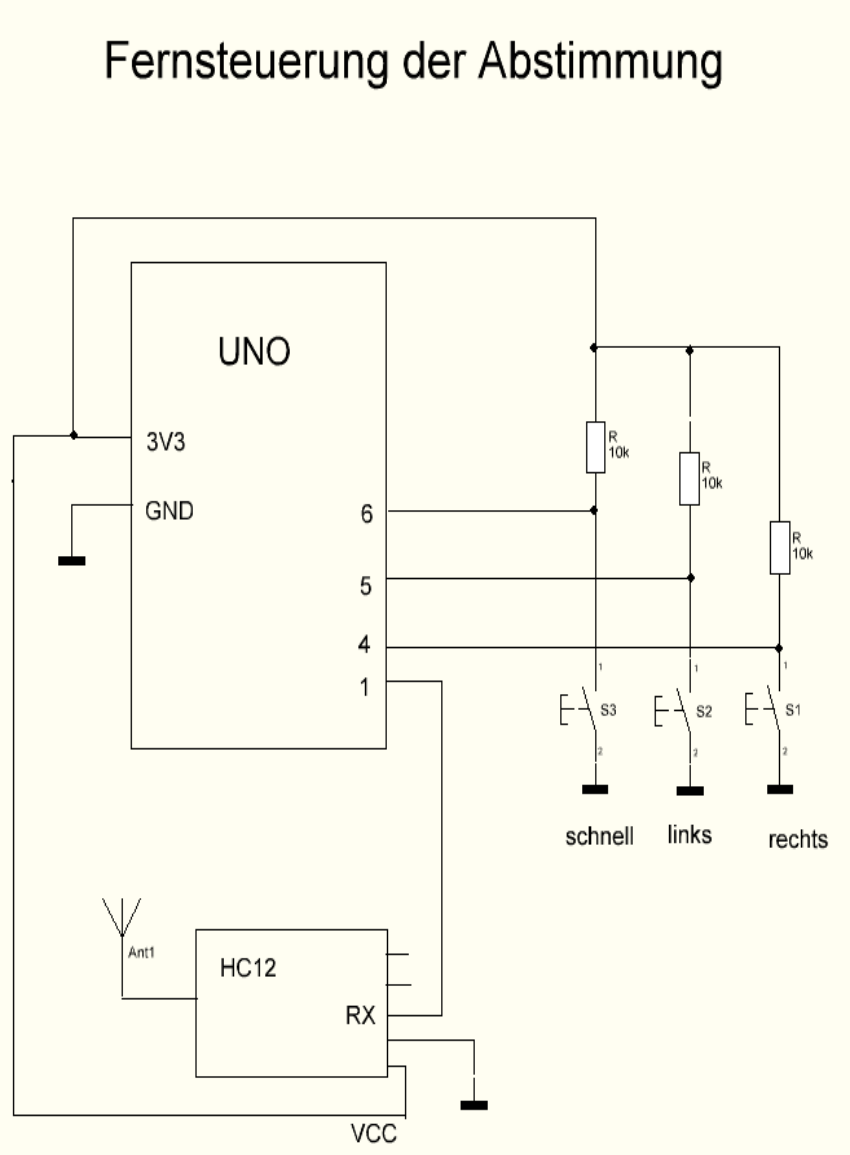
Fernsteuerung für einen Abstimm-Motor

Eine einfache Schaltung zur Steuerung eines DC-Motors.
Ich entwickelte sie, um damit meine fernabstimmbare Empfangsloop zu bedienen. Im Shack sind drei Taster. Drückt manS1, dann dreht der Moter rechts herum. Mit der Taste 2 nach links. Und drückt man dazu noch die dritte Taste, dann dreht der Motor schneller. Und natürlich kann man diese Sache noch erweitern und z.B. eine Bandumschaltung vorsehen.

Der Sender:



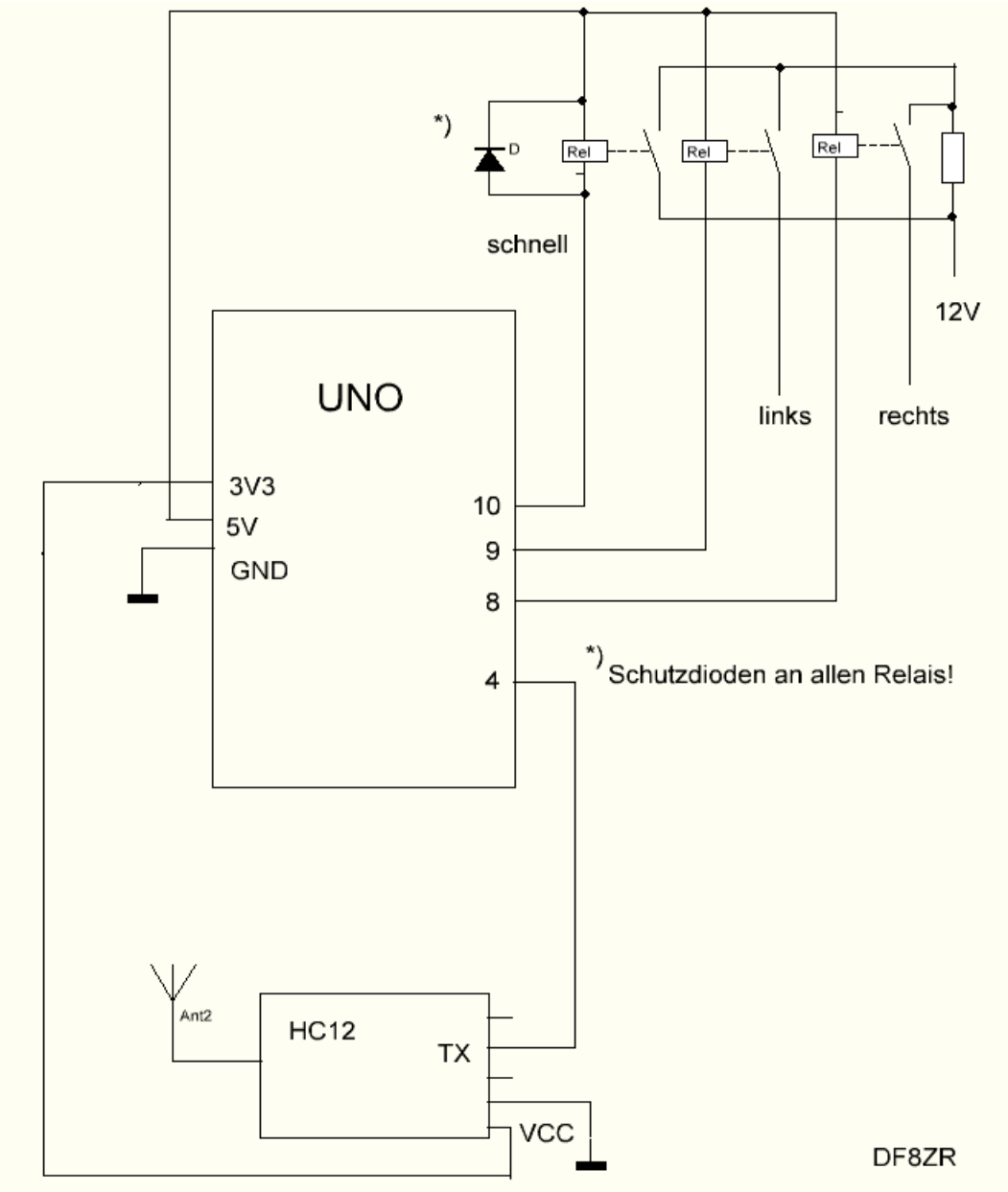
Man kann auch andere Arduinos einsetzen. Der Mikroprozessor wird über das Koaxkabel gespeist. Ebenso die aktive Elektronik

der Empfangsantenne. Den Motor sollte man gut entstören, damit keine Geräusche beim Abstimmen aufkommen. Denn die optimale Stellung hört man am Nutzsignal. Setzt man die Antennenleistung des HC12 auf P1, dann kann man auch das Koaxkabel als Übertragungsstrecke verwenden. Damit ist alles ziemlich gut gegen Fremdsignale abgeschirmt. Hier nun die Software:

Der Empfänger

Er arbeitet mit Relais, denn die Stromaufnahme kann das Schaltvermögen des Arduinos überschreiten. Man verwende 5V-DC-Relais. Den Mikro sollte man mit Freilaufdioden schützen. Ob man nun den schnellen Lauf mit der hier dargestellten Überbrückung eines Reihenwiderstandes macht oder andere Quellen schaltet, ist dem Anwender überlassen. Doch eines möchte ich noch erwähnen:

Die Schaltung wurde **nicht** für eine Sende-Antenne, z.B. passive Loop, entworfen. Hier spielen ganz andere Energien in den Mikro hinein. Der würde Sendeleistungen von mehr als 1W sicherlich nicht überleben. Dafür nimmt man besser eine solide Drahtverbindung.



Nachfolgend die Software:

Es wurde eine etwas aufwendige Lösung gefunden. Man könnte für jede Kennung einen Pin des Arduinos als Eingang eines Tasters dem Sender zuordnen. Da ich aber ursprünglich nur drei Tasten vorgesehen hatte, bilde ich erst das Muster des Outputs am Empfänger aus drei Kennungen. Ich sende die Werte 1,2und3. Wenn ich den 4. Kanal aktivieren möchte, sende ich hintereinander eine 1 und eine 3. Die Kennungen werden als char über die serielle Schnittstelle mit TX auf den RX des HC12 gegeben. Dann sendet der. Wird kein Taster gedrückt, sendet der

HC12 auch nicht. Das schont die Stromquelle und den Baustein, Der wird nämlich warm beim Dauersenden. Will man die Sendeenergie über das Koaxkabel mitsenden, dann muss neben der üblichen Entkopplung von der Gleichstromversorgung und des Signalpfades vermutlich eine Leistungsdämpfung vorgenommen werden. U.U. Werden sonst bei +10dBm schon die Bauteile der übrigen Elektronik zerstört.

Das Prinzip ist erweiterungsfähig. Ich plane noch eine Ergänzung für einen 5. Kanal, über den ich das Empfangsband umschalten möchte. Dafür spendiere ich aber einen weiteren Abfrage-Pin am Arduino. Das ist bei der Komplexität dann doch übersichtlicher.

```
 /* HC12senden
senden von Vor, Back und Quick
es gibt fünf Schaltzustände: alles aus, langsam nach rechts, schnell nach rechts,
langsam nach links und schnell nach links. Wenn man nicht möchte, dass die Realais "klappern", dann
muss manin einer angemessenen Reaktionszeit die Schaltzustände an den Outputs beständig darstellen.
Ich bilde hier dazu ein nachtriggerbares, monostabiles FlipFlop mit der Software nach.
Bei mehr als 4 Zuständen wird das aber auf Kosten der Reaktionszeit gehen!
*/
int a =0;
int b = 0;
int c = 0;
int d = 0;

bool senden = false;

void setup() {

  Serial.begin(9600); // Öffnet die serielle Schnittstelle bei 9600 Bit/s:

bool senden = false;

int ledPin = 9;
int vorPin = 4;
int backPin = 5;
//const int quickPin = 6;
int quickPin = 6;

  pinMode(ledPin,OUTPUT);
  pinMode(vorPin,INPUT);
```

```
}


void loop() {

senden = false;
Serial.end();

int vorPin = 4;
 pinMode(vorPin,INPUT);

 int backPin = 5;
  pinMode(backPin,INPUT);

int quickPin = 6;
  pinMode(quickPin,INPUT);

int ledPin=9;

pinMode(ledPin,OUTPUT);



 //Abfrage auf  Taster Vor= rechts drehen

   if (digitalRead(vorPin)==LOW){


  // digitalWrite(9,LOW);

   a = 1;
    senden = true;
    //Serial.print(1,DEC);

  delay(20);
  // }  else{
//digitalWrite(9,HIGH);
   //}
   }


//Abfrage auf  Taster Back

   if (digitalRead(backPin)==LOW){
   b = 2;
   senden = true;
    //Serial.print(2,DEC);
  delay(20);
  }
```

```
  //Abfrage auf  Taster Quick

   if (digitalRead(quickPin)==LOW){
   c = 3;
   senden = true;
     delay(20);
   // Serial.print(3,DEC);

  }

Serial.begin(9600);

   d= a + b + c;

 while (senden){

if (d == 1) {                  //nur langsam nach rechts drehen
Serial.print(d,DEC);//1
 delay(10);
 a=0;
 d=0;//Stop senden
 senden=false;
Serial.end();
  }
 //d=2;
 if (d == 2 | senden){                    //nur langsam nach links drehen
 Serial.print(d,DEC);//2
 delay(20);
 b=0;
 d=0;//Stop senden
 senden=false;
 Serial.end();
}

if (d == 4 | senden) {   // d = 1 +3 = schnell nach rechts
  Serial.print(d,DEC);//3
 delay(20);
//d=0;//Stop senden
 Serial.print(3,DEC);
 delay(20);
 a=0;
 c=0;
 d=0;//Stop senden
 senden=false;
Serial.end();
}

if (d == 5 | senden) {   // d = 2 +3 = schnell nach links
  Serial.print(d,DEC);
 //delay(10);
```

```
 delay(20);
 b=0;
 c=0;
 d=0;//Stop senden
 senden=false;
Serial.end();
}
 }
senden = false;                    // die Whileschleife wird nur aktiv, wenn senden=true ist
d=0;                              //wenn keine Taste gedrückt wird, sendet HC12 nicht!
a=0;
b=0;
c=0;

  delay(10);                  //warten auf das Ende der Aufnahme und Abarbeitung im
//Empfänger

}
```

Der Empfänger

Wenn man eine primitive Lösung z.B. durch Senden von char und am Empfänger nachfolgendes Abfragen auf die Kennung realisiert, dann kann man bei gleichzeitiger Betätigung von zwei Tasten bereits in Schwierigkeiten kommen. Solche Lösungen liefern an den Outputs über die Empfangssoftware häufig keine reine Gleichspannung. Mit Wechselströmen arbeiten aber die Relais schlecht. Daher wird hier das Prinzip eines retriggerbaren, monostabilen Flip/Flops per Software nachgebildet. Innerhalb einer gewissen Zeit muss ein Nachtriggersignal empfangen werden, dann bleibt der Ausgang für das Relais ohne Unterbrechungen bestehen. Erst nach einer unterhalb der menschlichen Wahrnehmungsfähigkeit abgelaufenen Verzögerung wechselt das Signal wieder in den Grundzustand.

Die Logik ist aufwendig, aber funktioniert sicher. Ich habe die Schaltsignale am Oszillografen beobachtet. Es kommt alle 100ms ein positiver Impuls im durchgeschalteten Zustand durch. Er ist aber kaum 80uS lang und stört nicht. Die ganze Schaltung muss man ohnehin möglichst abgeschirmt einsetzen. Bei jeder digitalen

Elektronik gibt es Oberwellen aus den Impulsen, die den Radioempfang stören können.

Solange man Tasten drückt, werden die Relais mit einer anhaltenden Null(0) geschaltet.

Senderseitig kann man die Doppelfunktion(1+30der2+3) auch mit 4 Tasten schalten. Dabei wird z.B. das dritte Signal 3 über Dioden mit den Schnell-Lauf-Tasten logisch verknüpft. So hat man zwei Tasten für den langsamen Rechts-Links-Lauf und zwei Tasten z.B. darunter die beiden für die schnelle Drehzahl. Nimmt man einen DC-Motor mit Getriebe, machen sich die Reaktionszeiten kaum bemerkbar.

Die Software im Empfangsteil:

```
/*motor
 * Das Programm Motor empfängt Daten zur Steuerung
 * der Drehrichtung(1=rechts, 2 = links, 3 = schnell) solange man
 * die Tasten am Sender drückt; die Ausgänge steuern mit Null 0 die
 * Relais an. Bitte beim Programm sender nachlesen
 */
#include <SoftwareSerial.h>

SoftwareSerial HC12(4, 5); // HC-12 TX Pin, HC-12 RX Pin

int a = 0;
int b = 0;
int c = 0;
int d = 0;
bool daten = false;

void setup()
  {
  DDRB = B00111111;      // alle Bits 8 bis   als Ausgang

  PORTB = B00111111;     //alle Pins auf 0

  Serial.begin(9600);            // Serial port to computer
  HC12.begin(9600);              // Serial port to HC12

}
```

```
  void loop() {


int v=100;
b=0;
c=0;
d=0;
PORTB = B00111111; //alle Pins auf 0
daten=false;



  while (HC12.available()) {        // If HC-12 hat daten


    char a =(HC12.read());

 if (a == 49) {  //ist eine 1 = langsam rechts drehen

    //PORTB = B00111110; //Pin8 auf 0

    d=1;
    daten=true;

  }


 if (a == 50) {    //ist eine 2
    //PORTB = B00111101; //Pin9 auf 0 langsam links drehen
    d=2;

    daten=true;

  }


 if (a == 51) {   // ist eine 3 schnellnach rechts drehen
    // PORTB = B00111010; //Pin10 und Pin 8 auf 0
    d=3;

    daten=true;

  }


if (a == 52) {  //ist eine 4
    // PORTB = B0011100  1; //Pin 9 und Pin10 auf 0
    d=4;

    daten=true;
    //delay(10);
```

```
  }


if (a == 53) {  //ist eine 5
    // PORTB = B00111011; //Pin9 und Pin10 auf 0
    d=5;

    daten=true;

  }


    Serial.print(a); //TEST
  }


while(daten){
 //char buffer [3];

if (d==1){
  PORTB = B00111110; //Pin8 auf 0 langsam rechts drehen
d=0;

//char y = d;

//Serial.print(y);
delay(v);
}

if (d==2){
  PORTB = B00111101; //Pin9 auf 0 = langsam links drehen
d=0;

delay(v);
}

if (d==3){
  PORTB = B00111010; //Pin8 und Pin10 auf 0 = schnell rechts drehen
d=0;


//Serial.print(d);
delay(v);
}

if (d==5){
  PORTB = B00111001; //Pin9 und Pin 10 auf 0 = schnell links drehen
d=0;
//Serial.print(d);
delay(v);
}
```

```
daten=false;

}


b=0;
c=0;
d=0;
PORTB = B00111111; //alle Pins auf 0
daten=false;



//delay(100);

}
```

Viel Spaß beim Nachbau!
DF8ZR; im Dez.2020